Análisis de la degradación de las comunicaciones en algoritmos de cómputo científico en un Cloud privado

Pettoruti José E.[1], Rodriguez Ismael P. [1], Chichizola Franco [1], De Giusti Armando E. [1 2]

[1]Instituto de Investigación en Informática LIDI (III-LIDI),
Facultad de Informática, Universidad Nacional de La Plata, Argentina
[2]CONICET
{josep, ismael, francoch, degiusti}@lidi.info.unlp.edu.ar

Abstract. Este trabajo presenta un análisis de la degradación de la red virtualizada, en la ejecución de algoritmos paralelos, para entornos de cómputo científico en un Cloud privado. Se introducen los conceptos básicos de Virtualización y Cloud Computing, para luego desarrollar un análisis de la degradación que introduce cada controlador de red virtual, de acuerdo a la técnica de virtualización: emulada o paravirtualizada, en el contexto del Hypervisor KVM integrado con Eucalyptus. Las pruebas efectuadas han sido realizadas con una aplicación de pasaje mensajes MPI que mide el tiempo de comunicación para diferentes tamaños de mensaje.

Keywords: *Cluster, Virtualización, Cloud Computing, Eucalyptus, Red, Algoritmos paralelos, Cómputo Científico.*

1 Introducción y motivación

Los avances en las tecnologías de virtualización y cómputo distribuido dieron origen al paradigma de Cloud Computing. Este último, se presenta como una opción atractiva para ambientes de Cómputo Científico, dado que ofrece la posibilidad de generar entornos de ejecución paralelos (Clusters Virtuales) bajo demanda, de acuerdo a las diversas necesidades de los usuarios.

En trabajos previos [1][2], se realizó el despliegue de un Cloud Privado para entornos de Cómputo Científico donde se analizó, en distintos escenarios, el overhead introducido por la arquitectura. Los resultados obtenidos dan origen al propósito de este trabajo, en el que se desea determinar si existe una degradación en las comunicaciones virtualizadas, pues este sería un factor importante en el tiempo de ejecución de un algoritmo paralelo que utiliza pasaje de mensajes.

La estructura de este trabajo es la siguiente: en la Sección 2 se presenta el concepto de la tecnología de virtualización y sus técnicas. Además, se describe brevemente KVM, el Hypervisor elegido para el entorno de pruebas. A continuación, la Sección 3 introduce conceptos elementales de Cloud Computing y el software Eucalyptus, seleccionado para desplegar un Cloud privado. La Sección 4 describe el trabajo experimental realizado y los resultados obtenidos. Por último, en la Sección 5 se presentan las conclusiones y las líneas de trabajo futuras en relación a este trabajo.

2 Virtualización

La virtualización es una tecnología que ha impactado significativamente en las infraestructuras de las organizaciones en los últimos años, brindando los beneficios de aprovechar, consolidar y reducir los costos de la infraestructura, como así también simplificar la gestión de los recursos y un mayor control de los mismos [3]. Esta tecnología es un componente fundamental del paradigma Cloud Computing.

El termino virtualización refiere a la abstracción de los recursos de hardware de la arquitectura x86. Para lograr este objetivo, se introduce una capa de software llamada Hypervisor o Monitor de Máquina Virtual (VMM) la cual permite la abstracción entre el hardware de la máquina física subyacente (Host) y el Sistema Operativo (SO) de la máquina virtual (Guest) [4].

El Hypervisor controla y gestiona los principales recursos del hardware físico (CPU, Memoria, Red y Almacenamiento) permitiendo la ejecución concurrente de múltiples instancias de máquinas virtuales (VM), cada una con su propio SO, sobre un único hardware físico [5].

2.1 Técnicas de Virtualización

A continuación se describen las técnicas de virtualización más utilizadas en la actualidad [6][7][8]:

Paravirtualización. Es una de las técnicas más populares. Los recursos no son emulados sino que se acceden por medio de drivers livianos, los cuales ofrecen un mejor rendimiento. Esto requiere que los kernels de los SO de las VMs sean modificados para incluir las System Calls necesarias para trabajar con esta técnica.

Virtualización Completa. Esta técnica proporcionar una completa emulación del hardware físico y no requiere asistencia de hardware ni de determinadas Systems Calls en los kernels de los SO de las VMs. Sin embargo, la necesidad de brindar al SO de la VM una interfaz completa del hardware físico, tiene un costo muy importante en el rendimiento. Dicho costo puede ser mitigado mediante el uso de virtualización asistida por hardware.

Virtualización asistida por hardware. El soporte de virtualización proporcionado por el hardware, surge a partir que los proveedores Intel (Intel-VT) y AMD (AMD-V), incorporando extensiones de virtualización para la arquitectura x86 en sus procesadores. Esto permite que los Hypervisors adicionen a sus técnicas este soporte, brindando una solución optimizada de virtualización sin requerir modificaciones en el SO de las VMs.

2.2 KVM (Kernel-based Virtual Machine)

Es un proyecto Open Source de virtualización, basado en las tecnologías mencionadas en la sección 2.1 con soporte asistido por hardware, integrado con el kernel de Linux. Su primera versión se introduce en el kernel 2.6.20, en febrero de 2007 [8][9].

Dado que KVM fue diseñado para utilizar el soporte de virtualización asistida por hardware, el mismo requiere que el CPU soporte las extensiones Intel VT-X o AMD-V, y que las mismas se encuentren habilitadas para su funcionamiento.

2.2.1 Arquitectura de KVM

KVM, en su arquitectura, implementa una VM como un proceso normal de Linux, planificado por el scheduler estándar del SO, donde cada CPU virtual es un thread. Esto permite a KVM beneficiarse de todas las características del kernel [9].

Los dispositivos virtuales son emulados por una versión modificada de QEMU, que provee emulación de BIOS, bus PCI, bus USB, controladores de disco IDE/SCSI, interfaces de red, entre otros.

En la Figura 1, se puede visualizar la arquitectura KVM.

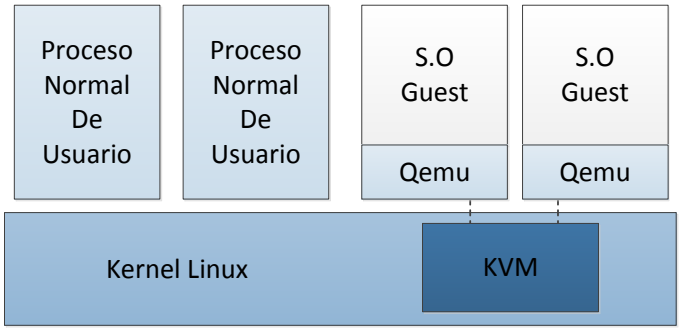

***Figura 1**: Arquitectura de KVM*

KVM soporta la *virtualización hibrida*, en la cual se utilizan drivers paravirtualizados instalados en SO Guest con el fin de permitir a las VMs un uso optimizado de los dispositivos de E/S sin necesidad de emularlos. Este soporte, en muchos casos presenta beneficios importantes en rendimiento.

2.2.2 Controladores de red

QEMU-KVM soporta un conjunto de controladores de red entre los cuales se pueden citar: *ne2k_pci, i82551, i82557b, i82559er, rtl8139, e1000, pcnet* y *virtio*. El driver *virtio* utiliza la técnica de Paravirtualización, todos los demás utilizan la técnica de Virtualización Completa (emulación de hardware).

En la actualidad los drivers más utilizados son *e1000* y *virtio*, dado que ambos brindan soporte para redes Gigabit y 10 Gigabit Ethernet.

Es importante destacar que el driver *virtio* permite la implementación de la virtualización híbrida.

3 Cloud Computing

Cloud Computing es un paradigma informático de cómputo distribuido, que proporciona grandes conjuntos de recursos virtuales (como ser hardware, plataformas de desarrollo, almacenamiento y/o aplicaciones), fácilmente accesibles y utilizables por medio de una interfaz de administración web. Estos recursos son proporcionados como servicios (del inglés, "as a service") y pueden ser dinámicamente reconfigurados para adaptarse a una carga de trabajo variable (escalabilidad), logrando una mejor utilización y evitando el sobre o sub dimensionamiento (elasticidad). El acceso a los recursos se realiza bajo la demanda de los usuarios, en base a un modelo de autoservicio [1][10].

3.1 Modelos de despliegue Cloud

Existen tres modelos de despliegue de un sistema de *Cloud Computing*: *Cloud público*, *Cloud privado* y *Cloud híbrido* [11][12][13].

Cloud público. Este modelo brinda la posibilidad de acceder inmediatamente a un conjunto de recursos con una mínima inversión, pero a su vez presenta un alto costo acumulado en el tiempo de uso del mismo y la ausencia de garantías sobre la privacidad y seguridad de la información del usuario.

Cloud privado. Tiene la capacidad de utilizar con mayor eficiencia la infraestructura física. Esto lo logra consolidando la infraestructura por medio de la virtualización. Además, brinda un alto nivel de seguridad sobre los datos sensibles de la organización por medio de políticas y medidas de seguridad establecidas en la Intranet de la organización.

Por otro lado, presenta una limitación en la escalabilidad en procesos de grandes demandas de servicios, dada por la capacidad de los recursos físicos. La misma se puede evitar con un modelo de Cloud híbrido.

Cloud híbrido. Este modelo permite a las organizaciones obtener los beneficios combinados de un Cloud privado y uno público, logrando así el aumento de la eficiencia en el uso de la infraestructura física, un mayor nivel de seguridad de los datos y una mayor disponibilidad de recursos para atender grandes demandas de servicio.

3.2 Características y beneficios de un Cloud

La utilización de un Cloud en Cómputo Científico presenta las siguientes características y beneficios [2]:

Recursos disponibles bajo demanda en el Cloud. En el caso de utilizar un Cloud público, no se debe afrontar la instalación y el mantenimiento de los recursos físicos, sólo se debe costear los servicios que se utilizan bajo demanda.

Por otro lado, los Cloud privados e híbridos, permiten optimizar la infraestructura existente consolidando la misma a través de la virtualización. Pueden existir diversas configuraciones de Clusters y ejecutar los mismos de acuerdo a la necesidad de los usuarios.

Escalabilidad y Elasticidad. Los recursos brindados por un Cloud son altamente escalables. Esto significa que una aplicación o un usuario pueden agregar o reducir dinámicamente sus recursos en respuesta a la variación en la carga de trabajo.

Aprovisionamiento automático de recursos. La distribución de los recursos se realiza de forma inmediata y automática sin requerir la intervención de personal técnico y/o administrativo. De esta forma, el personal técnico se libera de tareas repetitivas de configuración y atención a demandas, mientras que el usuario accede rápidamente a los recursos.

Autoservicio. Un usuario puede solicitar directamente los recursos que necesita, sirviéndose de los mismos por medio de una interfaz que le brinda control directo sobre el despliegue y configuración de los recursos, evitando así las demoras en el acceso a los mismos.

3.3 Software para despliegue de un Cloud: Eucalyptus

Eucalyptus es un proyecto de software Open Source bajo la licencia GPL, que permite implementar, administrar y ofrecer Infrastructura como Servicio (IaaS) sobre arquitecturas Clouds privadas e híbridas [13][14].

Eucalyptus surge como una alternativa a Amazon EC2 [15], brindando servicios Web compatibles con la API de Amazon; esto permite que las herramientas de gestión se utilicen en ambos entornos, ofreciendo compatibilidad y capacidad de migración entre Clouds.

Eucalyptus se presenta como un software para el despliegue y gestión de Clouds. El mismo puede ser instalado sobre las principales distribuciones de Linux, incluyendo Ubuntu, Red Hat Enterprise Linux, CentOS y Debian, sin la necesidad de alterar las instalaciones existentes del SO. Además, permite la utilización de diversos Hypervisors de virtualización, como ser KVM, XEN, entre otros.

4 Trabajo experimental

En trabajos previos [1] [2] se realizó el despliegue de un Cloud privado, del cual se utilizaron sus servicios de infraestructura (IaaS) para configurar un Cluster Virtual conformado por VMs [16][17]; y un Cluster Dedicado sobre los nodos físicos del Cloud. Ambos ambientes de cómputo han sido preparados para ejecutar aplicaciones científicas, que utilizan la librería OpenMPI para pasaje de mensajes.

La Tabla 1, presenta el hardware utilizado para el despliegue del Cloud.

***Tabla 1**: Hardware de despliegue.*

Cantidad	*Equipo*	*Formato*	*Procesadores*	*Memoria RAM*	*Disco Rigido*	*Red*
2	Dell PowerEdge R610	Rack	2 x Intel Quad-Core Xeon E5620 2.4 GHz	48 GB	500 GB	BCM5709 Gigabit Ethernet
1	Dell PowerEdge R710	Rack	2 x Intel Quad-Core Xeon E5520 2.27 GHz	24 GB	4 x 500 GB	BCM5709 Gigabit Ethernet

Una vez configurado el Cluster Virtual y el Cluster Dedicado, se procedió a realizar la ejecución de las siguientes aplicaciones científicas: una solución paralela al problema de N-Reinas [18] y algunas aplicaciones seleccionadas de la suite de benchmarks de sistemas paralelos NAS [19].

A partir de la ejecución de las aplicaciones mencionadas, se ha analizado el tiempo de ejecución de las mismas y se ha calculado el overhead introducido por la arquitectura Cloud.

Habiendo efectuado el cálculo del overhead y analizado los resultados, se detecta que el overhead introducido por la arquitectura es excesivo en aplicaciones que poseen una alta demanda de comunicación de red (se caracterizan por tener un tiempo de comunicación significativo sobre el tiempo de procesamiento), dando origen a los propósitos de este trabajo.

A continuación, se llevaron a cabo pruebas de comunicaciones utilizando una aplicación que hace uso de las librerías de pasajes de mensajes OpenMPI, con el objetivo de determinar si existe una degradación en la red virtualizada entre las VMs del Cloud con respecto a los nodos del Cluster en la red física subyacente.

El siguiente pseudocódigo corresponde a la aplicación Ping-Pong MPI para las pruebas de comunicaciones:

```
procesoPing{
  Sincronizar antes de enviar los datos
  for tamañoMensaje 1 to T (Potencias de 2)
    Realiza Pre-Envios para preparar las caché
    Tomar tiempo de inicio
    for cantPings 1 to P
      MPI_Send(buf,cantidadDeBytes,...)
      MPI_Recv(buf,cantidadDeBytes,...)
    Tomar tiempo de fin
    Calcula el tiempo promedio de comunicación
```

```
procesoPong
  Sincronizar antes de recibir los datos
  for tamañoMensaje 1 to T (Potencias de 2)
    Realiza Pre-Envios para preparar las caché
    for cantPings 1 to P
      MPI_Recv(buf,cantidadDeBytes,...)
      MPI_Send(buf,cantidadDeBytes,...)
```

Con el fin de determinar la degradación de la red virtualizada en la ejecución de aplicaciones de cómputo científico, se evaluaron los controladores de virtualización *e1000* y *virtio*, soportados por el Hypervisor KVM, efectuando las pruebas con la aplicación Ping-Pong MPI ya mencionada. Se utilizaron tamaños de mensaje entre 1 Byte y 1 GigaByte (potencias de 2), con 30 pre-envíos y 100 repeticiones para cada tamaño de mensaje.

Finalizadas las pruebas de comunicaciones y calculados los resultados, se concluyó que existe una degradación de la red virtualizada con respecto a la red física subyacente. La Figura 2, presenta un resumen de la degradación de la red virtualizada con los drivers *e1000* y *virtio*.

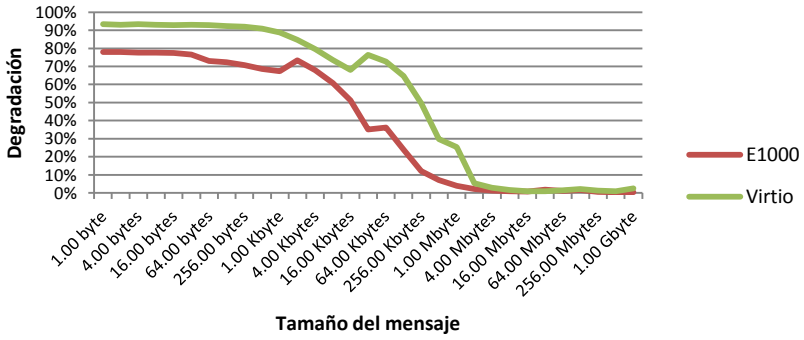

Figura 2: *Resumen de la degradación de la red virtualizada.*

Contemplando que en un entorno de computo científico para aplicaciones paralelas que utilizan pasaje de mensajes MPI, los tamaños de mensajes mayormente utilizados se ubican entre 1 byte y 64 Kbyte, se analizaron en detalle los tiempos de comunicación obtenidos en este intervalo. La Figura 3, refleja los resultados obtenidos.

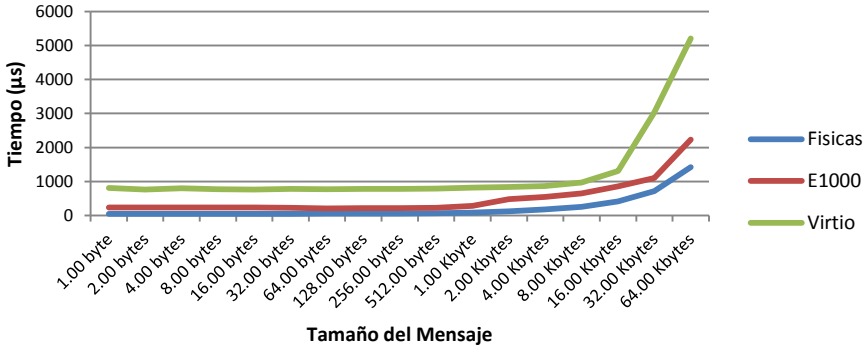

Figura 3: *Tiempos de comunicación para mensajes entre 1 byte y 64 Kbyte.*

A partir de este análisis, se puede visualizar que la degradación de la red virtualizada responde al overhead de latencia introducido por cada controlador de red.

Además, se realizaron pruebas de comunicación utilizando la herramienta ping del SO Linux, ejecutando en modo Flood con 10000 repeticiones y un tamaño de mensaje de 12 bytes, el cual es el tamaño mínimo permitido por la herramienta. Se han obtenido resultados que confirman las apreciaciones anteriores. En la figura 4, se pueden visualizar dichos resultados.

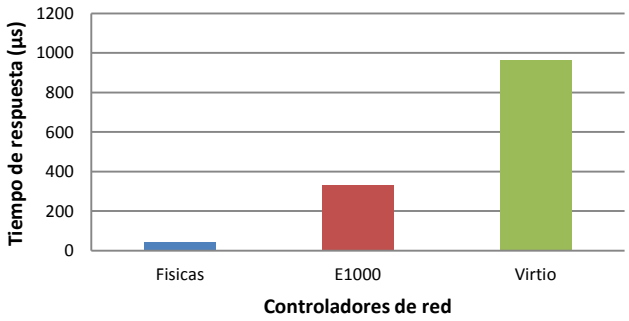

Figura 4: *Tiempos de respuestas herramienta ping SO Linux.*

5 Conclusiones y Trabajo Futuro

En base a todo el trabajo realizado, podemos concluir que para la ejecución de aplicaciones de cómputo científico con pasaje de mensajes MPI, el controlador de red virtualizada que menos degrada la comunicación es *e1000*. A pesar de ser un driver implementado con la técnica de virtualización completa, presenta un rendimiento superior al ofrecido por *virtio*, el cual es implementado con técnica de paravirtualización.

Además, del análisis efectuado, se evidencia que la virtualización de la red debe aún evolucionar para lograr un mejor rendimiento en la ejecución de algoritmos paralelos en entornos virtualizados.

Las líneas de trabajo futuro relacionadas con esta investigación incluyen la ejecución de las pruebas de comunicación con los controladores emulados y paravirtualizados en otros Hypervisors, como ser XEN y VMWare. También, es de gran interés analizar la implementación de dichos controladores buscando los orígenes de la degradación en la comunicación.

Por otro lado, es deseable estudiar el rendimiento de nuevas placas controladores de red, que incorporan soporte de virtualización en el mismo hardware.

Referencias

1. Rodriguez, I., Pettoruti, J.E., Chichizola, F., De Giusti, A.: Despliegue de un Cloud Privado para entornos de cómputo científico. In: Proceedings del XI Workshop de Procesamiento Distribuido y Paralelo (WPDP) - XVII Congreso Argentino de Ciencias de la Computación (CACIC 2011). La Plata, Argentina. (2011).
2. Pettoruti, J., Rodriguez, I.: Cloud Computing en aplicaciones científicas. Arquitectura, configuración y análisis experimental de costo/performance. Tesina de Grado, Facultad de Informática, UNLP. La Plata, Argentina. (2011).
3. History of Virtualization. http://www.vmware.com. Agosto 2011.
4. VMware: Understanding full virtualization, paravirtualization and hardware assist. Technical Report. http://www.vmware.com/resources/techresources/1008. USA. (2007).
5. Popek, G.J., Goldberg, R.P.: Formal Requirements for Virtualizable Third Generation Architectures. In: Communications in the ACM, Volume 17, Number 7, pp. 412--421. USA. (1974).
6. Adams, K., Agesen, O.: A comparison of software and hardware techniques for x86 virtualization. In: Twelfth International Conference on Architectural Support for Programming Languages and Operating Systems (2006).
7. VMware: Understanding full virtualization, paravirtualization and hardware assist. http://www.vmware.com/resources/techresources/1008. Reporte Técnico, (2007).
8. Nussbaum, L., Anhalt, F., Olivier, M., Gelas, J.: Linux-based virtualization for HPC clusters. In: Montreal Linux Symposium (2009), pp. 221—234. Canada. (2009).
9. KVM – Kernel Based Virtual Machine. Red Hat. White Paper. http://www.redhat.com. (2009).
10. Vaquero, L. M., Rodero-Merino, L., Caceres, J., Lindner, M.: A Break in the Clouds: Towards a Cloud Definition. In: ACM SIGCOMM Computer Communication Review, Volume 39, Issue 1, pp. 50--55. USA. (2009).
11. Chen, X., Wills, G. B., Gilbert, L., Bacigalupo, D.: TeciRes Report: Using Cloud for Research: a Technical Review. In: Computing, pp. 1--52. UK. (2010).
12. Eucalyptus User's Guide (2.0). http://open.eucalyptus.com. Junio 2012.
13. Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., Zagorodnov, D.: The Eucalyptus Open-Source Cloud-Computing System. In: 9th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGRID '09), pp. 124-131, IEEE Computer Society, Washington (2009).
14. Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., Zagorodnov, D.: Eucalyptus: A Technical Report on an Elastic Utility Computing

Archietcture Linking Your Programs to Useful Systems. Technical Report, UCSB Computer Science Technical Report Number 2008-10. Santa Barbara, USA. (2008).
15. Amazon Elastic Compute Cloud (Amazon EC2). http://aws.amazon.com/ec2/. Junio 2012.
16. Eucalyptus Systems: Eucalyptus® Cloud Computing Platform User Guide Version 1.6 (2010).
17. Eucalyptus Systems: Eucalyptus® Cloud Computing Platform User Guide Version 1.6 (2010).
18. Bruen, A., Dixon, R.: The n-queens problem. In: Discrete mathematics, volume 12, pp. 393--395. (1975).
19. Bailey, D., Harris, T., Saphir, W., van der Wijngaart, R., Woo, A., Yarrow, M.:The NAS Parallel Benchmarks 2.0. Technical Report NAS-95-020. USA. (1995).